Long-Term Experiments with an Adaptive Spherical View Representation for Navigation in Changing Environments

Feras Dayoub, Grzegorz Cielniak and Tom Duckett

School of Computer Science, University of Lincoln, LN6 7TS Lincoln, United Kingdom
fdayoub,gcielniak,tduckett@lincoln.ac.uk

Abstract

Real-world environments such as houses and offices change over time, meaning that a mobile robot's map will become out of date. In this work, we introduce a method to update the reference views in a hybrid metric-topological map so that a mobile robot can continue to localize itself in a changing environment. The updating mechanism, based on the multi-store model of human memory, incorporates a spherical metric representation of the observed visual features for each node in the map, which enables the robot to estimate its heading and navigate using multi-view geometry, as well as representing the local 3D geometry of the environment. A series of experiments demonstrate the persistence performance of the proposed system in real changing environments, including analysis of the long-term stability.

Keywords: Persistent Mapping, Omnidirectional Vision, Mobile Robot Navigation.

1. INTRODUCTION

Maintaining an up to date representation of the surrounding environment is a necessity for mobile robots to be able to work with people in their everyday environment and to have the ability to localize and navigate using sensory information. Most work in mobile robot mapping considers only how to acquire the initial representation of the environment, but there has been little work on how to update the map during long-term operation in changing environments.

An examination of the literature on visual mobile robot localization and mapping reveals two main approaches: metric methods [1, 2], which aim to estimate and track the absolute position of a robot within a geometric map, and appearance-based topological methods [3, 4], which represent the environment as a graph where the nodes of this graph correspond to places in the real environment. Between these two main branches, there is a hybrid approach [5]. In a hybrid map the environment is typically represented by a global topological map which connects local metric maps. The motivation for this hybrid type comes from the complementary strengths and weaknesses of metric and topological methods. Full metric maps do not scale well to large-scale environments, while pure topological maps cannot be used for accurate navigation inside a node. The existing approaches both topological and metric require a certain level of stability (static world assumption). This leads to problems when applying these methods in our everyday environments where we tend to change the appearance of our surroundings by adding, removing or changing the arrangement of objects, which implies that the localization and mapping methods must have flexibility, robustness and adaptation ability, along with some level of geometric accuracy.

In this paper, we propose a method to update the reference views of a hybrid metric-topological map for a changing environment, where the environment is represented as an adjacency graph of nodes on a topological level, and each node on the metric level of the map represents the 3D location of image features on a sphere, as shown in Fig. 1. The spherical representation of the nodes creates a connection between the topological and metric level of the map. A group of image features is used as a qualitative descriptor for global localization on the topological level, and the 3D location of these features on the sphere is used for estimating the heading needed for the navigation system at the metric level. In this way the proposed representation gives a balanced solution between the accuracy of geometric maps and the flexibility of topological maps.

The remainder of this paper is structured as follows. After discussing related work in Section 2, we introduce our memory model and the adaptation mechanism in Section 3. In Section 4, we describe the heading estimation method for navigation using the map. Section 5 explains how to update the spherical views of the map using the memory model. Section 6 presents the experiments and results obtained. Finally, we draw conclusions and discuss future work in Section 7.

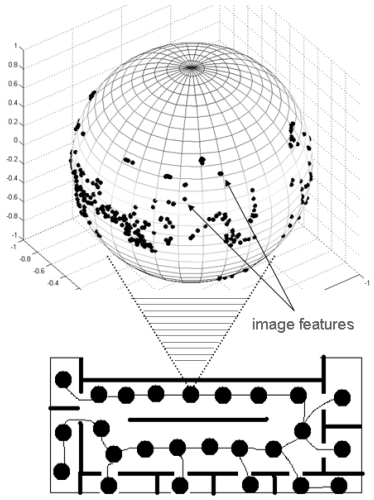

Figure 1: Proposed hybrid metric-topological map. The environment is represented as an adjacency graph of nodes on a topological level and each node on the metric level of the map represents the 3D location of image features on a unit sphere. Our method represents the direction of the features from the centre of the sphere, which corresponds to the centre of that node.

2. Related Work

Most previous approaches to mapping in dynamic environments assume that the underlying structure of the environment is static, and try to separate moving objects from the stationary parts. The dynamic effects are often treated merely as measurement outliers. Alternatively moving objects are detected and tracked separately using multi-target tracking techniques [6]. Another recent approach tries to classify landmarks as moving or stationary, and incorporates reversible data association within a sliding window of recent observations, to allow moving objects to be included into the Simultaneous Localization and Mapping (SLAM) estimate [7]. In general, while these approaches mitigate some problems of classical SLAM algorithms, they cannot handle long-term changes to the structure of an environment.

Several authors have investigated mapping systems that incorporate simple forgetting mechanisms based on recency weighting. For example, Yamauchi and Beer [8] developed a system that learns and updates the topology of a map during runtime based on successful and unsuccessful attempts to traverse a given link. Andrade-Cetto and Sanfeliu [9] developed an EKF-based mapping system that is able to forget landmarks that have disappeared, where an existence state associated with each landmark measures how often it has been seen. However, none of these methods are general enough to handle environmental changes occurring at different rates, nor has the long-term robustness of these approaches been demonstrated in real world environments.

This paper tackles the problem of working in a changing environment by using an appearance based approach to represent the workplace of the robot. An early approach for appearance-based mapping and localization was published in [10] where the operational area of the robot is represented as a graph. The nodes in this graph represent distinctive places and the edges represent the transitions between places. This approach consists of two stages. In the first stage the robot is driven through its operational area to learn a model of the environment, by taking a sequence of images in certain places and then creating a map from these images. In the second stage the robot uses the map to find its current position, i.e. the node which is most similar to the current view and also estimate its heading.

Two different approaches to measure the similarity between images have been presented in the literature: global and local methods. The global methods capture global properties of the image using approaches based on colour histograms [11], principal component analysis (PCA) [12], Fourier transform [13], etc. The local methods extract local properties from the image and produce a group of landmark features. Local features including SIFT [14], SURF [15], MSER [16], etc., are used to find the similarity between images. The local methods have been shown to be more reliable and robust to illumination and viewpoint changes, thanks to the feature descriptors that are built using a local region around selected feature points. Each feature is described by a high-dimensional vector, which has high invariance to image translation, scaling and rotation, and partial invariance to illumination changes and affine projection.

The similarity between two images can be measured using the number of feature correspondences between the two images. Matching of individual features can be done by finding the closest feature in the feature descriptor

space. However, the method can be time consuming if the number of images in the map is large and the search is done linearly. Using a text retrieval approach, Sivic and Zisserman [17] presented a very fast retrieval system using a visual vocabulary. Nister and Stewenius [18] extended the ability of the system by using hierarchical K-means clustering to improve the performance, so that Fraundorfer et al. [19] were able to implement global localisation in real-time.

Different methods have been introduced to enable a mobile robot equipped with a visual topological map to navigate. Recently, Goedeme et al. [20] presented a system based on wide-baseline image features matching, which enables the robot to follow a pre-recorded sequence of omnidirectional images. Guerrero et al. [21] presented a hierarchal localization system, which uses 1D three view geometry to achieve local metric localization which can be used to navigate the robot. Booij et al. [22] built their navigation system based on heading estimation to achieve a hill-climbing behaviour with no need for a pre-recorded path to follow.

This work introduces a complementary component for mobile robot mapping and localization systems that use features extracted from images to represent the appearance and/or geometry of places in the map. The main aspect of the previous works on vision-based navigation that is superceded by our approach is the ability to adapt the stored reference views of the robot's map in response to environmental changes during long-term operation. Note that we do not address the issue of autonomous exploration and map acquisition - it is assumed in our work that the robot has already acquired an initial map using a state-of-the-art method (see, for example, the approach of Ila et al. [23] in which nodes and links are added to the map with regards to information content), and we also do not address the problem of topological changes in this work. The main contribution of this paper is to introduce a principled approach for combining updates to the appearance and geometry of the stored reference views of the robot's map, with extensive long-term experiments demonstrating the robustness of the approach.

3. Long-term adaptation using memory stores

We introduce a method to update the reference views of a robot's map in a changing environment. A naive solution to this problem would be simply to replace the image representation for each node in the topological

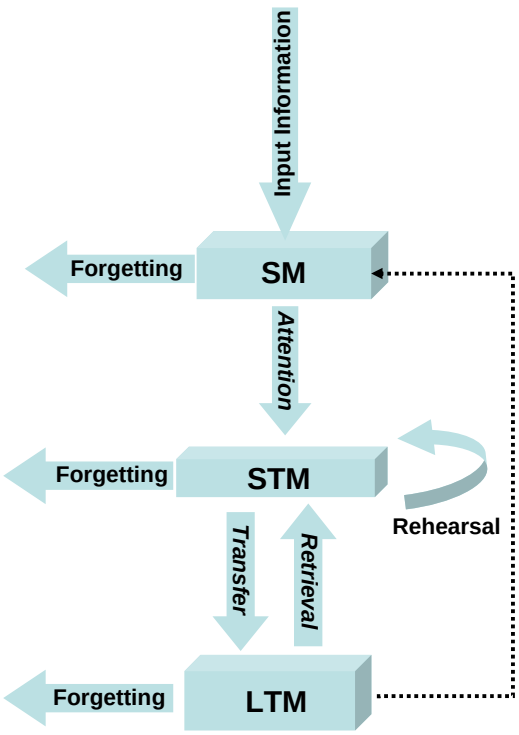

Figure 2: The Information Processing Model. SM: Sensory memory. STM: Short-term memory. LTM: Long-term memory.

map from time to time, in order to reflect the changed appearance of the corresponding location. Provided that the robot is correctly localized, this approach would enable the robot to remove out-of-date information from the map. However, such an approach could also remove useful features due to temporary occlusions, and could lead to catastrophic results in the case of localization errors. A better solution would be to update the image representation of a node incrementally, by gradually adding information about new stable features in the environment, while removing information about features that no longer exist. In order to achieve this we adopted short-term and long-term memory concepts based on the multi-store model of human memory proposed by Atkinson and Shiffrin [24]. This model, which forms the basis of modern memory theories, divides human memory into three stores (see also Fig. 2):

- sensory memory (SM),
- short-term memory (STM),
- long-term memory (LTM).

The sensory memory contains information perceived by the senses, and selective attention determines what information moves from sensory memory to short-term memory. Through the process of rehearsal, information in STM can be committed to LTM to be retained for longer periods of time. In return, the knowledge stored in LTM affects our perception of the world, and influences what information we attend to in the environment.

Applying these concepts to our approach for robot mapping, the sensory memory will contain the features extracted from the current image. Then an attentional mechanism selects which information to move to STM, which is used as an intermediate store where new observations are kept for a short time. Over this time the system uses a rehearsal mechanism to select features that are more stable for transfer to LTM. In order to limit the overall storage requirements and adapt to changes in the environment, the system also contains a recall mechanism that forgets unused feature points in LTM by removing these features from the node. LTM is used in turn by the attentional mechanism for selecting the new sensory information to update the map. Thus in our approach only the *changes* to the mapped environment are selected for input to STM.

Algorithm 1 Update the reference view.

Definitions:
CrrNode: The reference view of the current node.
CrrView: The current view for the current node.
CrrSTM: The current STM for the current node.
STMlng: The maximum number of states in the STM.
LTMlng: The maximum number of states in the LTM.
newFP: The difference between the CrrView and CrrNode.

---

for (every visit to the node) {
    newFP = **recall**(CrrNode , CurrView , LTMlng)
    rehearse(CrrSTM, newFP , STMlng)
}

3.1. Recall, Rehearsal, Transfer

We model the world as a set of discrete places. In our experiments, omni-directional vision is used to provide the features for localization and mapping. We assume that an initial map of the whole environment has already been created by the robot, e.g. using an existing algorithm for topological mapping of static environments. (In this work we selected the places by hand.)

Each place has two memory stores: STM, which is a temporary stage, and LTM, which provides the reference views in the map used for self-localization. We assume that the robot is able to self-localize by matching features extracted from the current view to the stored reference views. The purpose of our algorithm presented here is to maintain up-to-date reference views for the nodes in the map, using recall and rehearsal concepts inspired by human memory.

To initialise the map, the image data from the robot's first tour of the environment is used. One image is selected to represent each node in the map. For each node, local features are extracted and used directly to initialise LTM, while STM for each node is initially assigned to be empty.

Thereafter, every time the robot visits an existing node, the following steps are carried out. Feature points are extracted from the current view. Self-localization is carried out by comparing the current features to the reference features of each node (LTM) to estimate the current node. In our case, we apply global localization by place

recognition using a simple winner-take-all strategy, although any appropriate self-localization algorithm could be applied, e.g. Markov localization. After localization, the current features are used in the recall stage for updating the LTM of the current node. Only new features which do not match any feature in LTM are used in the rehearsal stage. Algorithm 1 describes the two main stages; (1) recall, where the difference in appearance between the reference and current views is computed, and (2) rehearsal, where this difference is used to update STM and commit persistent new features of the location to LTM.

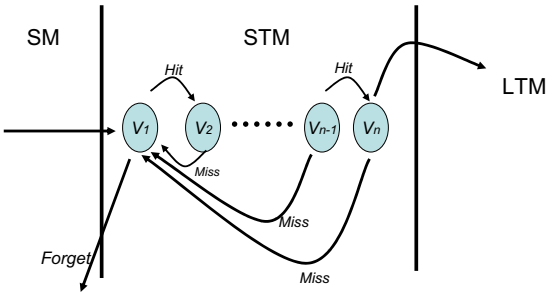

Figure 3: The rehearsal stage in the STM.

Algorithm 2 The rehearsal stage in the STM.

```
for (every feature in CrrSTM ){
    if (feature in newFP){
      Move the feature to the next state.
      if (feature state > STMlng){
        Move the feature to the CrrNode.
        Remove the feature from CrrSTM.
      }
      else if (feature in the first state){
            Remove the feature from CrrSTM.
      else
            Reset the feature to the first state.
      }
    }
}
for (every feature in newFP ){
    if (feature was not in CrrSTM){
      Add the feature to CrrSTM in the first state.
    }
}
```

Algorithm 2 shows the rehearsal process for a stored feature in STM, which is also represented as a finite state machine in Fig. 3. This stage represents what Atkinson and Schiffrin called rehearsal in their memory model (Fig. 2), i.e. the process of continually recalling information into the STM in order to memorise it. In order to transfer a feature point from STM to LTM the feature has to be seen frequently in that node. Features enter STM from sensory memory and must progress through several intermediate states (V_1 to V_n) before transfer to LTM. Every time the robot visits the node and finds the feature ("hit"), the state of the feature is moved closer to LTM. However if the feature is missing from the current view ("miss"), it is returned to the first state (V_1) or forgotten if it is already there. This policy means that spurious features should be quickly forgotten, while persistent features will be transferred to LTM.

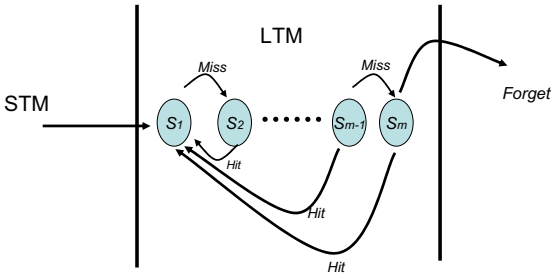

Figure 4: The recall stage in the LTM.

Algorithm 3 The recall stage in the LTM.

```
newFP = []
for (every feature in CrrNode) {
    if (feature in CrrView){
      Reset the feature to the first state.
    else
      Move the feature to the next state.
    }
    if (feature state > LTMlng) {
      Remove the feature from CrrNode.
    }
}
for (every feature in CrrView) {
    if (feature not in CrrNode) {
      Add the feature to newFP.
    }
}
return newFP
```

Algorithm 3 shows the recall process for a stored feature in LTM, which is also represented as finite state machine in Fig. 4. This process first involves updating the LTM by matching the reference view to the current view. In order to remain in the LTM, a feature has to be seen occasionally in that node. In contrast to rehearsal, features enter LTM from STM and must progress through several intermediate states (S_1 to S_m) before being forgotten. Stored features which have been seen in the current view are reset to the first state (S_1), while the state of features which have not been seen is progressed, and a feature point that passes through all states without a "hit" is forgotten. Finally, recall returns the list of new features that were not already present in the LTM.(i.e the difference in appearance between the current and reference views).

4. Heading estimation using the spherical views

For navigation, the robot needs not only to estimate its current location in the topological map, but also its orientation with respect to the current node. We estimate the heading of the robot relative to the current node, by estimating the relative orientation of the current view of the robot with respect to the stored reference view. In this way, the reference views function as way-points that the robot can use to travel from one place to the next. In our case, the desired heading is estimated using the epipolar geometry for spherical cameras [25]. The model of the spherical camera consists of a unit sphere whose centre is the centre of the curved mirror. The omnidirectional camera we are using can be treated as a spherical camera because it is a central omnidirectional camera. A central omnidirectional camera has a point called an optical centre, on which all projection rays meet. So after calibrating the camera using the toolbox by Scaramuzza et al. [26], each reference image is represented by a spherical view (Fig. 1) where the feature points are projected on a unit sphere.

Given two spherical views with centres C and C', a scene point P can be back-projected through the two spheres to the centre of projection for each camera. Let X represent the position of P in the reference frame of C

and x represent the projection of X on the sphere, then we can write:

$$\lambda x = X, \quad \lambda \in \mathbb{R}_+ \tag{1}$$

In the same way for the second camera we will have:

$$\lambda' x' = X', \quad \lambda' \in \mathbb{R}_+ \tag{2}$$

where x', X' are the 3D points in the frame of $C^{'}$. Assuming that $C = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ with $\mathbf{R}$ and $\mathbf{T}$ expressing the transform of the camera coordinates between C and $C^{'}$, we can write:

$$X' = \mathbf{R}X + \mathbf{T}, \tag{3}$$

then by substituting Eqs. 1 and 2 into Eq. 3, we get:

$$\lambda' x' = \lambda \mathbf{R} x + \mathbf{T}. \tag{4}$$

which leads to the following relation based on the epipolar geometry for spherical cameras:

$$(x')^T \mathbf{E} x = 0, \tag{5}$$

where $\mathbf{E}$ is the essential matrix. This matrix can be linearly solved using eight pairs (or more) of corresponding points from the two spheres [27]. In our case, the corresponding points are generated from the two views using the descriptors of the image features which will typically generate more than 8 correspondences between the two views. For this reason and the fact that the false matches will always be part of the matching process, using the RANSAC algorithm [28] is a very efficient way to minimize the effect of the outliers and find the best essential matrix to fit most of the points.

The robot in our case is working on a planar floor which means that the rotation between the spherical views will only be around the vertical axis. Using this fact, we can simplify the process of estimating the essential matrix by restricting it to the following sparse form [29], assuming translation in x-y plane and rotation around z-axis:

$$\mathbf{E} = \begin{bmatrix} 0 & 0 & e_{13} \\ 0 & 0 & e_{23} \\ e_{31} & e_{32} & 0 \end{bmatrix}. \tag{6}$$

Based on the method introduced by Hartley and Zisserman in [30], the essential matrix is factored to give Eq. 7 which contains the rotation matrix $\mathbf{R} \in SO(3)$ and the skew-symmetric matrix $[\mathbf{T}]_\times$ of the translation vector $\mathbf{T} \in \mathbb{R}^3$:

$$\mathbf{E} = [\mathbf{T}]_\times \mathbf{R}. \tag{7}$$

This will generate multiple solutions, i.e. four possible combinations of $\mathbf{T}$ and $\mathbf{R}$. However, by applying the positive depth constraint we obtain the one solution where the reconstructed point lies outside of the two spheres [31].

After the estimation of $\mathbf{T}$ and $\mathbf{R}$, the robot can find the heading toward the reference view, α, using the translation direction $\mathbf{T}$:

$$\alpha = atan(\mathbf{T}[y], \mathbf{T}[x]), \tag{8}$$

and also the rotation between the current view and the reference view, γ, using the rotation matrix:

$$\mathbf{R} = \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{9}$$

5. Updating the spherical views

Updating the reference views of the map based on the proposed memory model means removing old unused features and adding new features during long-term operation of the robot. So in order to preserve the ability to use the updated spherical views for heading estimation, the feature points which need to be moved to the STM and LTM stores of each node should be located on the reference sphere as if these features were seen from the same point where the node was first created. This ensures that the robot will keep the ability to use the reference views for heading estimation and therefore navigate using the map.

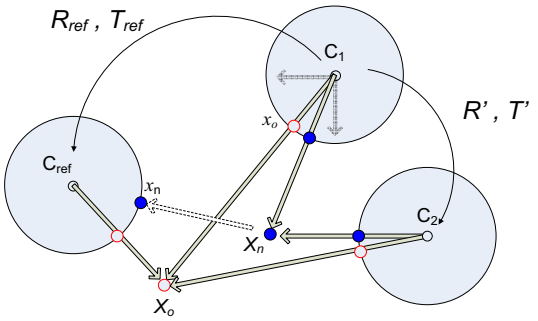

Figure 5: Reference view updating. The current view C_1 is matched with the previous view C_2 and the reference view C_{ref} to estimate the coordinates of new features in the spherical representation of the reference view.

In order to achieve this, we reconstruct the 3D position of feature points shared between one view from the current visit and one view from the previous visit to the node. The current and previous views are each obtained by selecting the image in the recorded sequence for that visit with the highest similarity score to the reference view. The 3D position of the shared points can be determined to unknown scale as the norm of the translation vector is fixed to unity. These points are divided into three groups: the points which already exist in the LTM store of the node, the points which already exist in the STM store of the node and the new points which need to be added to the STM.

In order to add these new features to the STM into their correct position on the sphere we use a simplified version of what is known in the computer vision literature as multibaseline stereo [32]. In our case, we only use two stereo pairs between three views: the reference view the current view and the view from the previous visit to the node. The views are captured in different visits to the node and we are not interested in recovering a 3-D map for a large scene; instead we want to update a single spherical view by adding new feature points to it. Linear triangulation is used to obtain the desired 3D position of a point. More details of the linear triangulation approach can be found in [30].

In Fig. 5, let X_o be one of the reconstructed positions for an image feature which is shared between the three views (C_1, C_2, C_{ref}) where C_{ref} is the reference view of the node and C_1, C_2 are the views from the current visit and the previous one, respectively.

Based on the stereo views (C_1, C_2), we can write:

$$X_o^{C_2} = \lambda_2 x_o, \tag{10}$$

where λ_2 is the depth of X_o based on the unknown scale of the stereo views (C_1, C_2), $X_o^{C_2}$ is the representation of X_o in the reference frame of C_1 and x_o is the projection of $X_o^{C_2}$ on the unit sphere of C_1.

Also, in the reference frame of the view C_1 and based on the stereo views (C_1, C_{ref}), we can write:

$$X_o^{ref} = \lambda_{ref} x_o, \tag{11}$$

where λ_{ref} is the depth of X_o based on the unknown scale of the stereo pair (C_1, C_{ref}) and X_o^{ref} is the representation of X_o in the reference frame of C_1.

Eqs. 10 and 11 mean that a point X_o shared between the three views will have different values $(X_o^{ref}, X_o^{C_2})$ depending on the scale of the reconstruction. This also means that we can convert between the different unknown scales:

$$X_o^{ref} = s X_o^{C_2}, \quad s \in R. \tag{12}$$

The value of s is estimated such that it minimizes the distance error between the 3-D point's correspondences between the two stereo pairs (C_1, C_2) and (C_1, C_{ref}). Outliers are rejected using robust statistics [33].

Now, let X_n be a reconstructed position of an image feature shared between the two views (C_1, C_2) but which does not exist in the view C_{ref}. In order to find the projection of X_n on the sphere of C_{ref}, first we need to convert to the scale of the stereo pair (C_1, C_{ref}) using s:

$$X_n^{ref} = s X_n^{C_2}, \tag{13}$$

where $X_n^{C_2}$ is the representation of X_n in the reference frame of C_1 based on the scale of the stereo views (C_1, C_2) and X_n^{ref} is the representation of X_n in the reference frame of C_1 based on the scale of the stereo views (C_1, C_{ref}).

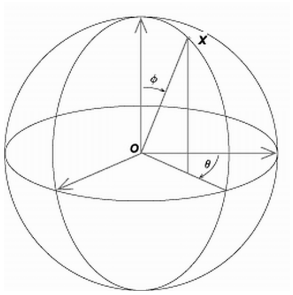

Figure 6: Spherical coordinates.

Then, as shown in Fig. 5, the view C_1 and the reference view C_{ref} are related by a rigid body displacement represented by the rotation matrix $\mathbf{R}_{ref} \in SO(3)$ and the translation $\mathbf{T}_{ref} \in \mathbb{R}^3$. We can transform X_n^{ref} to the frame of the reference view C_{ref} as follows:

$$X_n^{C_1} = \mathbf{R}_{ref} X_n^{ref} + \mathbf{T}_{ref}. \tag{14}$$

Finally, the position of the new feature in the STM store of the reference view sphere, x_n, can be found by normalization:

$$x_n = \frac{X_n^{C_1}}{\|X_n^{C_1}\|}. \tag{15}$$

5.1. Recursive Filtering

A key issue for any long-term mapping and localization system is the danger that the interdependency between mapping and localization introduces a positive feedback loop, where measurement noise may be picked up and amplified to the point where the system becomes unstable (see related discussion in [34]). An important question is therefore how to avoid such instabilities and maintain robust long-term performance. In our hybrid mapping system, new features will not be recorded in exactly the same location as the original features, so they have to be projected into the spherical view representation, as described above. As the multi-view geometry calculations will be sensitive to noise, the process of adding new points could introduce an accumulated error which would eventually result in the degradation of map quality. To mitigate these effects, we can exploit the fact that the position of each new added feature can be re-estimated repeatedly during rehearsal and recall, meaning that we can apply recursive filtering methods to produce estimates that will tend to be closer to the true values of the measurements. So, if we represent the position of the point by its spherical coordinates, (see Fig. 6), as

$$\mathbf{x} = [\theta, \phi]^T,$$

then we can formulate the problem as an estimation problem with $\mathbf{x}$ as the state of the system.

It is clear that the process model for such system is simple where the state is unchanged as soon as it is been added to the STM. Whereas the measurement for this state will come as 3D points on the sphere and the observation model which relates these measurements to the state vector will involve the following nonlinear mapping:

$$\mathbf{z} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \sin(\theta)\sin(\phi) \\ \cos(\theta)\sin(\phi) \\ \cos(\phi) \end{bmatrix}, \tag{16}$$

where z is the observation vector.

Due to this nonlinearity, we can consider this as a simple nonlinear estimation problem and therefore use any suitable technique such as the Unscented Kalman Filter (UKF) [35] in which a small number of carefully chosen sample points is propagated in each estimation step. For a state space with dimension L, $2L + 1$ points are selected such that their sample mean is the state vector and their covariance is the process covariance (so in our experiments, 5 points were used). The nonlinear function is applied to each point in turn to yield a cloud of transformed points which provide a compact parameterization of the underlying distribution. We chose this filter over other nonlinear filters such as the Extended Kalman Filter (EKF) [36] due to its more accurate estimation properties.

Figure 7: The experimental platform. An ActivMedia P3-AT robot equipped with an omnidirectional vision system.

6. Experiments and Results

The following experiments were designed to evaluate the system:

1. to test the accuracy of the system in responding to changes, we conducted a laboratory experiment where the environment was changed manually;

2. to test the robustness of the system to degradation of map quality over very long-term operation, we conducted a experiment where we artificially looped a dataset multiple times; and

3. to test the overall performance of the system under real operating conditions, we conducted an experiment using data collected from a real changing environment over a period of approximately two months.

Our experimental platform was an ActivMedia P3-AT robot equipped with a GigE progressive camera (Jai TMC-4100GE, 4.2 megapixels) with a curved mirror from 0-360.com (see Fig. 7). For local feature extraction we use the SURF algorithm. This algorithm extracts local features from the scale-space of the image based on the Hessian matrix and approximates the second order Gaussian derivatives with box filters. A fast non-maximum suppression algorithm is also used. The resulting algorithm has a good performance in the extraction process and a high accuracy. For more details, see [15].

6.1. Testing the system using data from a manually changed environment

This experiment was carried out in our robotics lab where we collected a set of images by driving the robot in a loop. The images were generated from 50 tours over a period of 3 days resulting in 1385 images. After each tour the appearance of the lab was changed manually. The changes involved the arrangement of the objects inside the room, including adding new objects like boxes and posters, removing existing objects individually and also covering them with movable office dividers for certain periods. Fig. 10 shows two images taken from the same node at different times.

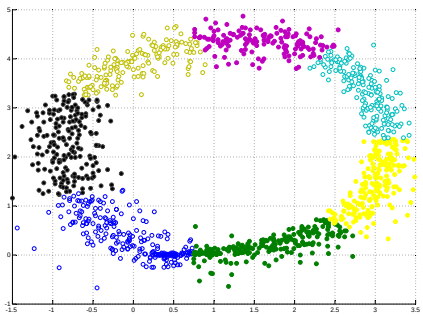

Figure 8: Ground truth positions of the recorded images obtained from the laser-corrected odometry. The constructed map consists of seven nodes, each colour representing the group of images which belong to the same node.

In order to obtain the ground truth positions for the collected data, the starting points for all 50 tours were initialized from a fixed place inside the room while each image was recorded along with a laser scan and odometry. This enable us to use LODO [37], a library for laser-based correction of odometry, attaching each image with a 2D position and a rotation relative to the starting point.

The first tour was used to create the topological map which consists of 7 nodes (selected manually). Fig. 8 shows the ground truth positions of the images, each colour representing the group of images belonging to the same node. The rest of the image sequence was used as input to the localization system. We used global localisation based on place recognition using a similarity score between the current and the reference views (winner-takes-all) as a first stage to locate the robot in one of the nodes [38].

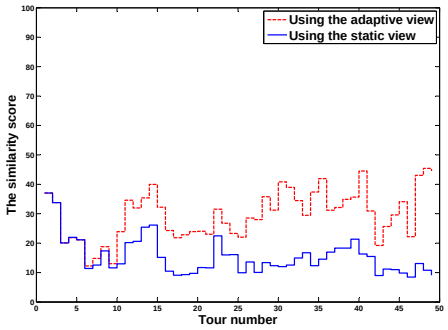

Figure 9: The similarity score for node 4 using the static view and the adaptive view.

To find the similarity score between two groups of feature points, we use the number of corresponding features M_{ij} between the two groups based on a nearest neighbour (NN) matching scheme using the value 0.7 as a threshold between the nearest and second-nearest neighbour, following [15]. The matching is carried out as follows. A feature point in the current view is compared to a feature point in the reference image by calculating the Euclidean distance between their descriptor vectors. A matching pair is detected, if its distance is closer than 0.7 times the distance of the second nearest neighbour. The similarity score between view C_j and a reference view C_i can be defined as:

$$S_{ij} = \frac{M_{ij}}{K_i} * 100, \tag{17}$$

where K_i is the number of features in the reference view V_i.

During a visit to a node in the map, the robot will capture a number of images as it goes through the node. Among these images the image with the highest similarity score is used to represent the view of the node for that visit and it is then used to update the reference view using the proposed updating mechanism.

Fig. 9 shows how the similarity score changed over the 49 tours for node number 4 in the map. As shown, using the adaptive views gave a higher similarity score, while for the static view the similarity score sometimes dropped below 10% as in tour 17. After the global localization step, the reference view of the node and the input image was used to estimate the rotation between the two views using Eq. 9, and then the estimated rotation was compared with the ground truth obtained from the laser-stabilized odometry. Using the sequence of collected images we repeated the same experiment once using static reference views for the map and then using the multi-store memory model both with and without the Unscented Kalman Filter. The static reference views were created

Table 1: The first experiment results

	Comparison measures		
Measure	Static Map	Adaptive Map without UKF	Adaptive Map with UKF
Error in estimating the rotation	$4.2^o \pm 4.1^o$	$4.5^o \pm 4.6^o$	$4.0^o \pm 4.1^o$
Mean number of matched points	81.8 ± 43.8	118.3 ± 54.4	120.8 ± 55.9
Number of matched points > 50	77.0%	95.1%	95.1%

Figure 10: Two panoramic views from the same place in the laboratory environment at different times.

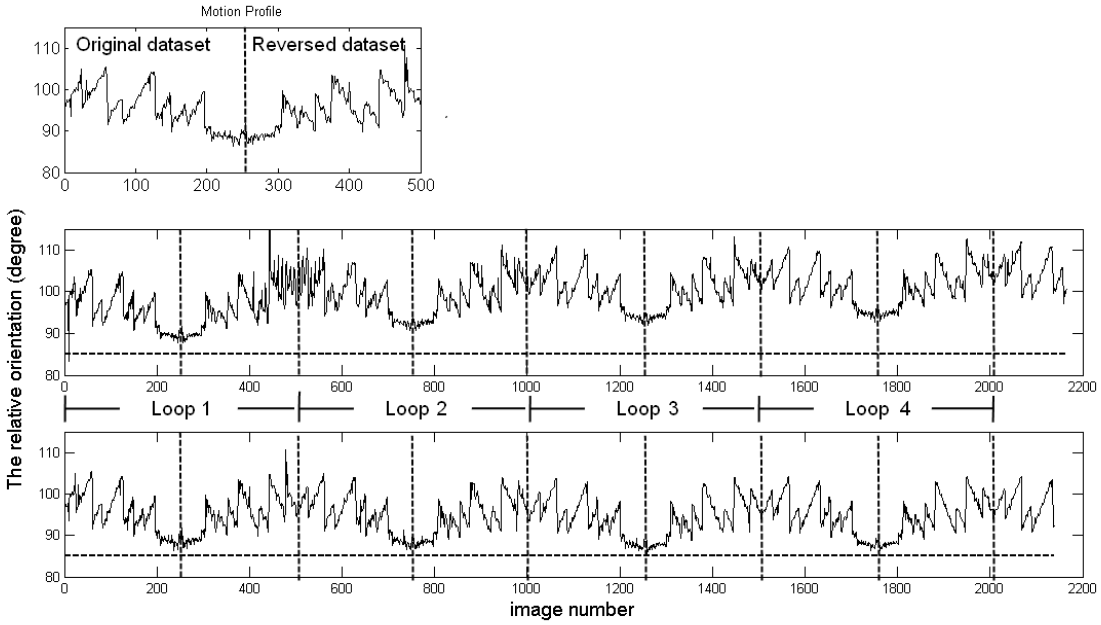

Figure 11: The top row shows the estimated heading relative to the reference view, using the original dataset combined with the reversed dataset. The second row shows how the estimation of the heading without the UKF drifts over time due to noisy measurements, where the combined dataset was looped repeatedly. The last row shows the effect of the UKF where the long-term drift is greatly reduced.

from the first run (similarly the first run was used to initialize LTM as described in Section 2) and the subsequent runs were used for localization.

Table 1 shows a comparison using several performance measures, showing mean and standard deviation, between the static and the adaptive approach using 8 stages for the LTM and 3 stages for the STM. As shown in the first row, the error in the estimation of the rotation did not drop significantly, whereas the average number of matched points for the winning node, which is used for the global localization, increased noticeably when we used the memory model, as shown in the second row. As the environment changed over time, the number of matched points for the winning node became smaller when the static reference views were used. As shown in the third row, during global localization the winning number of matched points was over 50 in 77.0% of cases for the static map and 95.1% for the adaptive memory model. In this relatively short-term experiment, we have found no significant difference in performance when using the Unscented Kalman Filter.

6.2. Testing the long-term stability of the system

In order to test the long-term stability of the system when updating the features on the spherical view representation, the following experiment was designed. First we took a dataset consisting of an image sequence recorded while we manually made changes to the environment. The robot first took a reference view in the middle of a room then it was placed with approximately 90^o rotation with respect to that reference view. Then the robot was moved

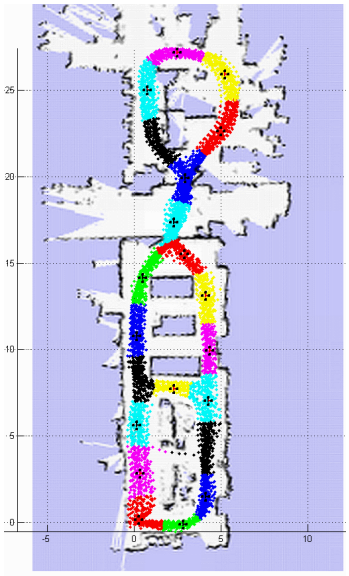

Figure 12: Ground truth positions of the recorded images obtained from the laser-corrected odometry in the office environment. The constructed map consists of 21 nodes, where each colour represents the group of images which were assigned to the same node. The crosses represent the reference views for each node.

back and forth covering approximately 1 m on each side of the location where the reference view was recorded. While the robot was moving, a set of 251 images was recorded to capture the changing appearance of the room.

Then we made a copy of this dataset, reversed the sequence of images in the copy, and appended the copy to the original dataset. The result of this operation is a single dataset representing one run of the robot forwards and backwards through the environment (see Fig. 11, top). This operation means that measurements errors in one direction should be balanced by equal and opposite errors in the opposite direction, so that the net drift in the estimated orientation of the robot after one such run should be zero (thus providing the reference standard for this experiment, i.e. an accumulated error of zero would represent a "perfect" result after a large number of repetitions of the same run). Finally the obtained dataset was "looped" or repeated multiple times (see Fig. 11), and used to test the updating mechanism for a single map node. By considering each image in the extended sequence as a daily visit to the node, the total number of visits was around 2200, which is approximately equivalent to a period of 72 months.

We can judge the stability of the proposed updating mechanism based on the relative orientation between the recorded images and the reference view in the LTM. The second row of Fig. 11 shows the rotation sequence estimated from around 2200 images but without the filtering step. The effect of the accumulated error is obvious here, making the motion profile drift gradually over successive loops of the dataset. (We also added a horizontal dashed line to this figure to make the drift more clear.) The third row shows the effect of the Unscented Kalman Filter step introduced in Section 5.1 where the drift is noticeably reduced.

6.3. Long-term experiment using data from a real changing environment

This experiment was carried out in a real changing environment of an office floor at the University of Lincoln. This area is used for student and staff activities. Over a period of approximately two months, we took the robot on tours between the offices while recording a set of images. The result was 36 tours (one tour per day) generating 6161 images with approximately 35cm between consecutive images. We used the first tour to create a topological map by choosing 21 omni-directional images from different places to form the reference views for the nodes. To do so, we adopted a two step technique. We start by clustering the images in a number of regions with fixed range (2.5m in our experiment) based on the geometric distance. In the second step we choose an image which is most similar to all the images in each region to be a reference view. Fig. 12 shows the ground truth positions of the recorded images. Each colour represents the group of images which were assigned to the same node. The crosses represent the reference view for each node. In order to obtain the ground truth data we used the GMapping library [39]. The GMapping algorithm provides a Simultaneous Localization and Mapping (SLAM) solution based on a Rao-Blackwellized particle filter. The output of the algorithm is an estimate of the robot trajectory along with an occupancy grid map of the environment. The first tour was used for creating the topological map. For each subsequent tour, we estimated the robot's true trajectory through the environment, then we aligned the

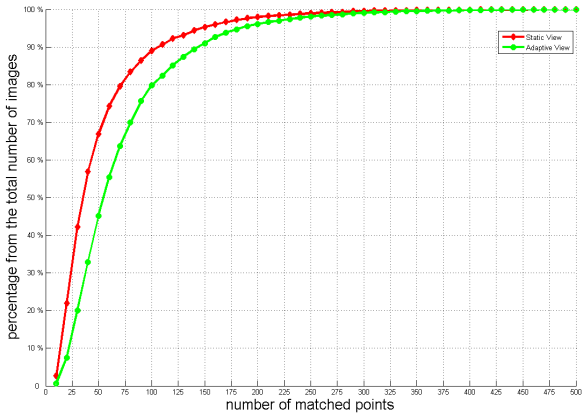

Figure 13: The cumulative percentage of matched points used for the global localization. The red line illustrates the results when the static reference views were used for the map whereas the green line corresponds to the adaptive views.

output map with the first tour map in order to transform the ground truth navigation trajectories into a common reference frame. We would like to emphasize that our method does not build or require a global metric map of the environment. We only use this information for ground-truth validation.

We tested our method for adapting the reference spheres in the map including the Unscented Kalman Filter using the collected images. Similar to the first experiment, we used a global localization method based on place recognition (winner-takes-all). We carried out the tests using the memory model with 8 stages in the LTM and 2 stages in the STM.

The results show that the failure rate of the global localization was around 1.5% when the adaptive approach was used whereas the failure rate for the static one was 3.5%. More importantly, the experiment demonstrates the ability to use the image features which match between the reference view and current image of the robot to estimate the appropriate heading. With the static views, the rotation could not be generated around 15% of the time, whereas the failure rate was only 5% for the adaptive views. This happens because the robot needs a sufficient number of matched features (more than 35) otherwise a degenerate solution for the essential matrix or a very noisy estimation for the heading could be obtained. The absolute angular error in the heading estimates was $9.6^o \pm 7.5^o$ for the static views and $9.3^o \pm 7.5^o$ for the multi-store memory model. As the results show, and due to the changing nature of the environment along with the occlusion effect, the multi-store memory model provides better representation for the appearance of the nodes leading to improved performance in global localization.

Fig. 13 shows the cumulative percentage of matched points used for the global localization. The red line illustrates the results when the static reference views were used for the map, whereas the green line corresponds to the memory model. The adaptive approach provides better matching scores leading to better localization performance and also better navigation performance. For example, 43% of the time the number of matched points between the reference view and the current image from the robot was below 35 points for the static views, whereas this happened only 24% of the time with the multi-store memory model.

7. Conclusions

This paper introduced a method to enable a mobile robot working in a non-static environment to update an internal representation of its environment in response to the changes in the appearance of that environment. The updating mechanism is based on long-term and short-term memory concepts, and uses local image features to update the reference views in a hybrid metric-topological map, while preserving the ability to use the updated reference views for heading estimation based on the multi-view geometry of spherical cameras. We proposed a spherical representation of the nodes as a connection between the topological and metric levels of the map. A group of image features is used as a qualitative descriptor for global localization on the topological level, and the 3D location of these features on the sphere is used for estimating the rotation angle needed for the navigation system at the metric level. In order to avoid the instabilities due to measurement noise and to maintain robust long-term performance, we used a recursive filtering step based on an Unscented Kalman Filter to update the metric level of the representation. The method was evaluated using data taken from real changing environments over a long period of time.

In this work we did not develop a complete system for autonomous navigation but we focused on one specific aspect, that is, how to update the reference views of the map in response to the changes in the appearance of the environment. Therefore we made the assumption that the map of the environment had already been built and that navigation on this map would use the reference views as way-points that the robot uses to travel from one place to the next. However, our experiments clearly demonstrate a significant improvement in both global localization and heading estimation during long-term operation using the proposed multi-store memory model.

One of the important parameters of the proposed memory model is the number of stages in both the STM and LTM. These parameters have a direct effect on the number of feature points inside each reference view, and they are influenced directly by the rate at which the environment is changing. In the case of a mainly static environment, the points will be concentrated on the first few stages of the LTM and both the rate of forgetting and the rate of adding new features from STM to LTM will be small. However for an extremely dynamic environment, the points will be distributed over all stages and the rate at which the points are transferred to and forgotten from the LTM will be high. Therefore the number of stages should be selected in a way that does not allow the features to be forgotten too quickly but at the same time keeping the number of points in the reference views at a reasonable size. In this work, the number of the stages in LTM and STM was determined empirically based on the recorded sensor data. In future work, the number of the memory stages could be learned depending on the dynamics of the real environment. The adaptive capability of the map could be further extended to the topological level, by making the robot able to add or remove nodes and links from the map.

8. References

[1] P. Elinas, R. Sim, J. J. Little, sigmaSLAM: Stereo vision SLAM using the Rao-Blackwellised particle filter and a novel mixture proposal distribution, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2006) 1564–1570.

[2] S. Se, D. Lowe, J. Little, Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks, The International Journal of Robotics Research 21 (8) (2002) 735.

[3] Z. Zivkovic, O. Booij, B. Kröse, From images to rooms, Robotics and Autonomous Systems 55 (5) (2007) 411–418.

[4] C. Valgren, A. Lilienthal, T. Duckett, Incremental Topological Mapping Using Omnidirectional Vision, in: Proc. IEEE International Conference on Intelligent Robots and Systems (IROS), 2006.

[5] B. Kuipers, Y. T. Byun, A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations, Toward Learning Robots. MIT Press, Cambridge, Massachusetts (1993) 4763.

[6] C.-W. Wang, C. Thorpe, S. Thrun, Online SLAM with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas, in: Proc. IEEE International Conference on Robotics and Automation (ICRA), Taipei, Taiwan, 2003, pp. 842–849.

[7] C. Bibby, I. Reid, Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association, in: Proc. Robotics: Science and Systems (RSS), Atlanta, GA, USA, 2007.

[8] B. Yamauchi, R. Beer, Spatial learning for navigation in dynamic environments, IEEE Trans. Systems, Man and Cybernetics 26 (3) (1996) 496–505.

[9] J. Andrade-Cetto, A. Sanfeliu, Concurrent map building and localization in indoor dynamic environments, Int. J. Pattern Recognition and Artificial Intelligence 16 (3) (2002) 361–374.

[10] I. Ulrich, I. Nourbakhsh, Appearance-based place recognition for topological localization, in: Proc. IEEE International Conference on Robotics and Automation (ICRA), 2000.

[11] H. Gross, A. Koenig, C. Schroeter, H. Boehme, Omnivision-based probabilistic self-localization for a mobile shopping assistant continued, in: Proc. IEEE International Conference on Intelligent Robots and Systems (IROS), 2003.

[12] N. Vlassis, B. Terwijn, B. Krose, Auxiliary particle filter robot localization from high-dimensional sensor observations, in: Proc. IEEE International Conference on Robotics and Automation (ICRA), 2002.

[13] E. Menegatti, M. Zoccarato, E. Pagello, H. Ishiguro, Image-based Monte Carlo localisation with omnidirectional images, Robotics and Autonomous Systems 48 (1) (2004) 17–30.

[14] D. Lowe, Object recognition from local scale-invariant features, in: Proc. IEEE International Conference on Computer Vision (ICCV), 1999.

[15] H. Bay, T. Tuytelaars, L. Van Gool, SURF: Speeded up robust features, in: Proc. European Conference on Computer Vision (ECCV), 2006.

[16] J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide-baseline stereo from maximally stable extremal regions, Image and Vision Computing 22 (10) (2004) 761–767.

[17] J. Sivic, A. Zisserman, Video Google: a text retrieval approach to object matching in videos, in: Proc. IEEE International Conference on Computer Vision (ICCV), 2003.

[18] D. Nister, H. Stewenius, Scalable recognition with a vocabulary tree, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.

[19] C. Fraundorfer, F. Engels, D. Nister, Topological mapping, localization and navigation using image collections, in: Proc. IEEE International Conference on Intelligent Robots and Systems (IROS), 2007.

[20] T. Goedemé, M. Nuttin, T. Tuytelaars, L. Van Gool, Omnidirectional Vision Based Topological Navigation, International Journal of Computer Vision 74 (3) (2007) 219–236.

[21] J. J. Guerrero, A. C. Murillo, C. Sags, Localization and matching using the planar trifocal tensor with bearing-only data, IEEE Transactions on Robotics 24 (2008) 494–501.

[22] O. Booij, B. Terwijn, Z. Zivkovic, B. Krose, Navigation using an appearance based topological map, Proc. of the IEEE International Conference on Robotics and Automation (ICRA).

[23] V. Ila, J. M. Porta, J. Andrade-Cetto, Information-based compact pose SLAM, IEEE Trans. Robot.

[24] R. Atkinson, R. Shiffrin, Human memory: A proposed system and its control processes, In K.W. Spence & J.T. Spence (Eds.), The Psychology of Learning and Motivation 2 (1968) 89–195.
[25] T. Akihiko, I. Atsushi, Multiple view geometry for spherical cameras, IEIC Technical Report (Institute of Electronics, Information and Communication Engineers) 105 (2005) 29–34.
[26] D. Scaramuzza, A. Martinelli, R. Siegwart, A toolbox for easily calibrating omnidirectional cameras, Proc. of the IEEE International Conference on Intelligent Systems, IROS06, Beijing, China.
[27] H. C. Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, Nature 293 (1981) 133–135.
[28] M. A. Fischler, R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (1981) 381–395.
[29] J. Košecká, F. Li, X. Yang, Global localization and relative positioning based on scale-invariant keypoints, Robotics and Autonomous Systems 52 (2005) 27–38.
[30] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd Edition, Cambridge University Press, 2004.
[31] B. K. Horn, Relative orientation, International Journal of Computer Vision 4 (1) (1990) 5978.
[32] S. B. Kang, R. Szeliski, 3-D scene data recovery using omnidirectional multibaseline stereo, International Journal of Computer Vision 25 (1997) 167–183.
[33] E. Bandari, N. Goldstein, I. Nesnas, M. Bajracharya, N. RIACS, Efficient calculation of absolute orientation with outlier rejection, BMVA Syposium on Spatiotemporal Image Processing.
[34] A. Ess, B. Leibe, K. Schindler, L. van Gool, A mobile vision system for robust multi-person tracking, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, USA, 2008.
[35] S. J. Julier, J. K. Uhlmann, A new extension of the kalman filter to nonlinear systems, in: Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Vol. 3, 1997, p. 26.
[36] J. K. Uhlmann, Simultaneous map building and localization for real time applications, transfer thesis, Univ. Oxford, Oxford, UK.
[37] A. Howard, Laser-stabilized odometry (lodo_driver), Available from http://robotics.usc.edu/~ahoward.
[38] F. Dayoub, T. Duckett, An adaptive appearance-based map for long-term topological localization of mobile robots, in: Proc. IEEE International Conference on Intelligent Robots and Systems (IROS), Nice, France, 22-26 Sept, 2008, pp. 3364–3369.
[39] G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with Rao-Blackwellized particle filters, IEEE Transactions on Robotics 23 (1) (2007) 34–46.

Feras Dayoub is a Ph.D. student at Centre for Vision and Robotics Research, Lincoln University, UK. He received his B.Sc. in Software Engineering from AlBaath University, Syria, in 2004. His research interests include mobile robotics, computer vision, and machine learning.

Grzegorz Cielniak is a lecturer in Computer Science at the University of Lincoln, UK. He obtained his Ph.D. in Computer Science from Örebro University, Sweden in 2007 and M.Sc. from Wroclaw University of Technology, Poland in 2000. The Ph.D thesis addresses real-time people tracking system for mobile robots. His research interests include mobile robotics, flying robots, AI, vision systems and people tracking.

Tom Duckett is a Reader in Computer Science at the University of Lincoln, UK, where he is also Director of the Centre for Vision and Robotics Research. He was formerly a docent (Associate Professor) at Örebro University, Sweden, within the Centre for Applied Autonomous Sensor Systems. He obtained his Ph.D. in Computer Science from Manchester University in 2001, M.Sc. with distinction in Knowledge Based Systems from Heriot-Watt University in 1995 and B.Sc. (Hons.) in Computer and Management Science from Warwick University in 1991, and has also studied at Karlsruhe and Bremen Universities. His research interests include autonomous robotics, computer vision, artificial intelligence, multi-sensor fusion, and food technology.